EUREKA: Dynamic Data Access with On-Chip Configuration Generation

Xinyu Niu and Wayne Luk
Imperial College London

Yu Wang
Tsinghua University

Outline

- motivation
- EURECA architecture
- case studies
- results
- summary

EURECA: key features

- routing challenge: dynamic data access applications
  - multiplexors close to memory, configuration distribution
  - on-chip configuration generation
  - cycle-by-cycle reconfiguration
- experimental results: VTR + Cadence
  - small area overhead: 1% of XC6V-SX475T FPGA
- Memcached, sparse matric vector, large-scale sorting
  - up to 1/15x design area
  - up to 2.2x clock speed

Static vs dynamic data access

- conditional arithmetic operators
- dynamic data access patterns

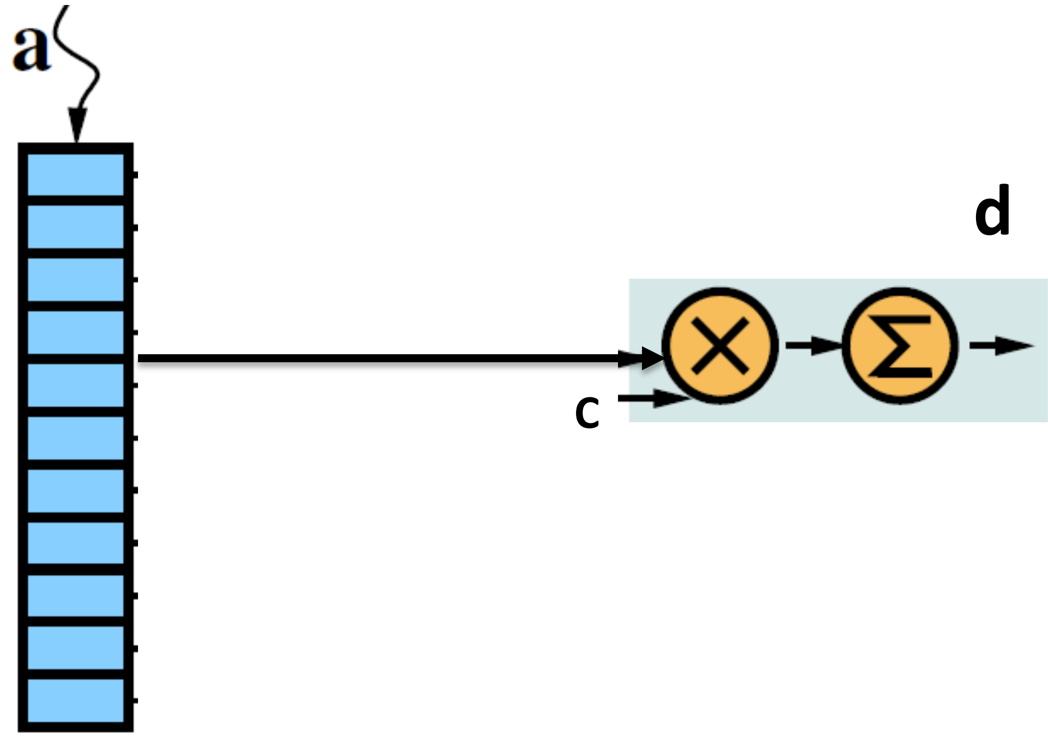
```
for (i=0; i<n; i+=N)
  #parallel unroll N
  for(j=0; j<N; j++){
     k = i*N + j;
     d[k] = a[k+1]  *  c[k];
}
```

static: easy

```
for (i=0; i<n; i+=N)
  #parallel unroll N
  for(j=0; j<N; j++){
     k = i*N + j;
     d[k] = a[b[k+1]]  *  c[k];
}
```

dynamic: hard

Example: dynamic data access

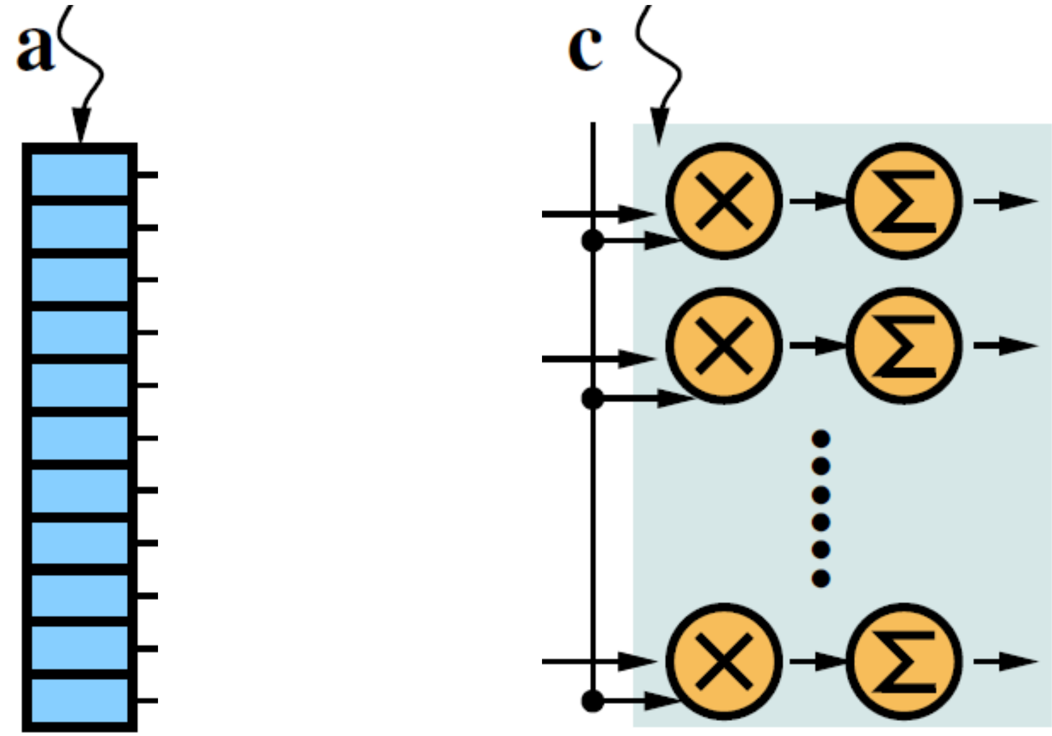



```
for (i=0; i<n; i++) {
    d[k] = a[b[k+1]]  *  c[k];
}
```

connections for
one data-path: easy

Example: dynamic data access

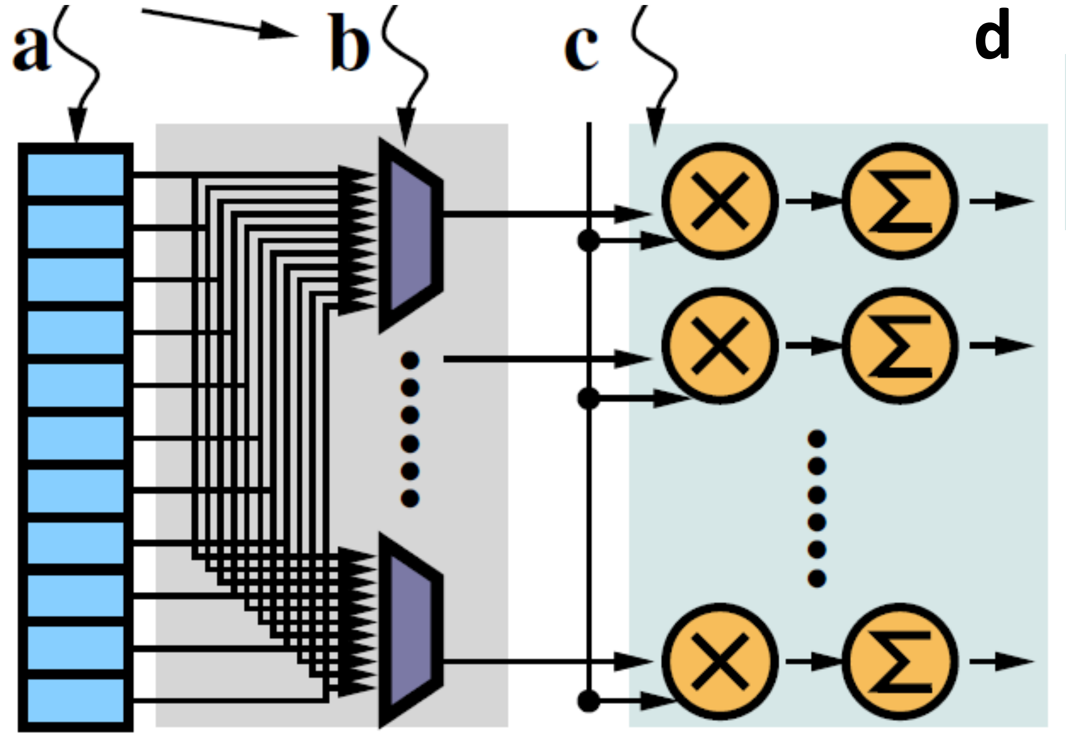
```
for (i=0; i<n; i+=N)
  #parallel unroll N=32
  for(j=0; j<32; j++){
     k = i*32 + j;
     d[k] = a[b[k+1]]  *  c[k];
}
```



connections for
32 data-paths: hard

Implementation 1: multiplexors

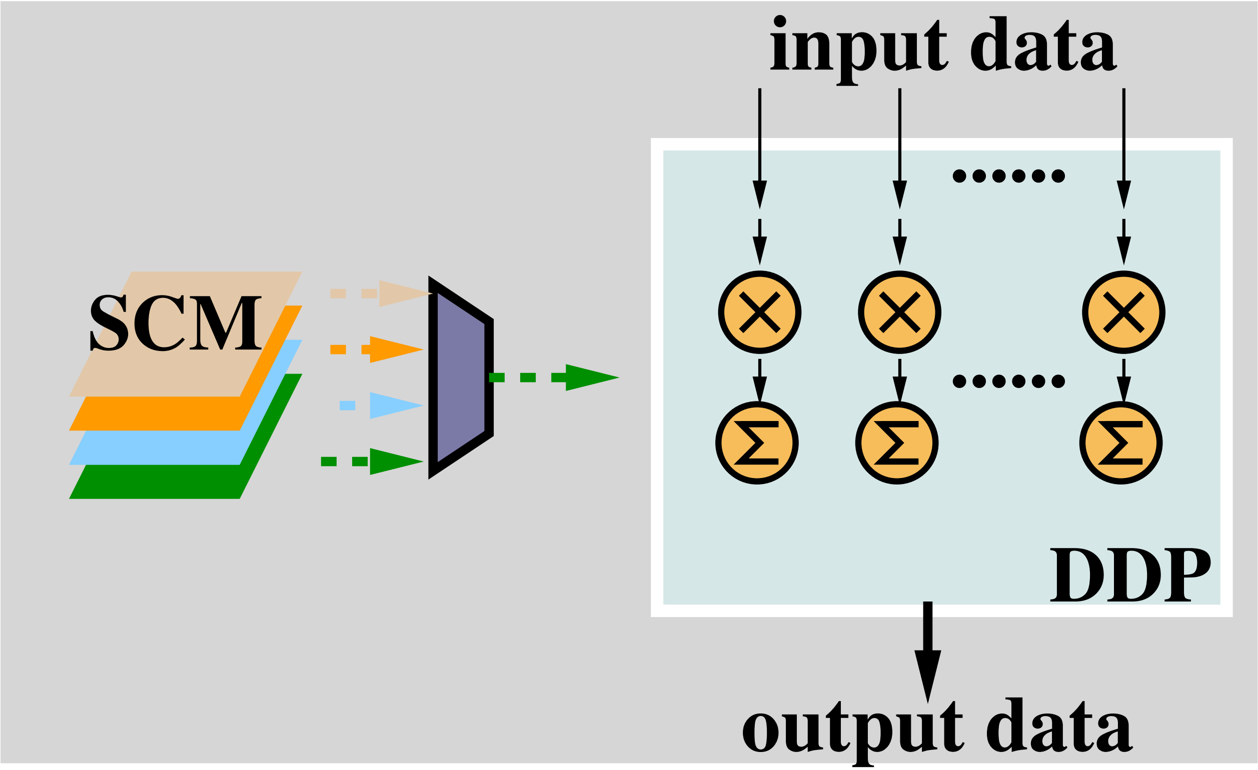
- congested routing
  - 1024-to-1024 bit connections
  - unroutable in XC6V-SX475T
- expensive user multiplexers



32 output ports, each 32-bit wide

Implementation 2: existing reconfigurable device

- need all possible configurations
- stored off-chip : partial reconfiguration: time overhead
- stored on-chip: multi-context FPGAs: area overhead

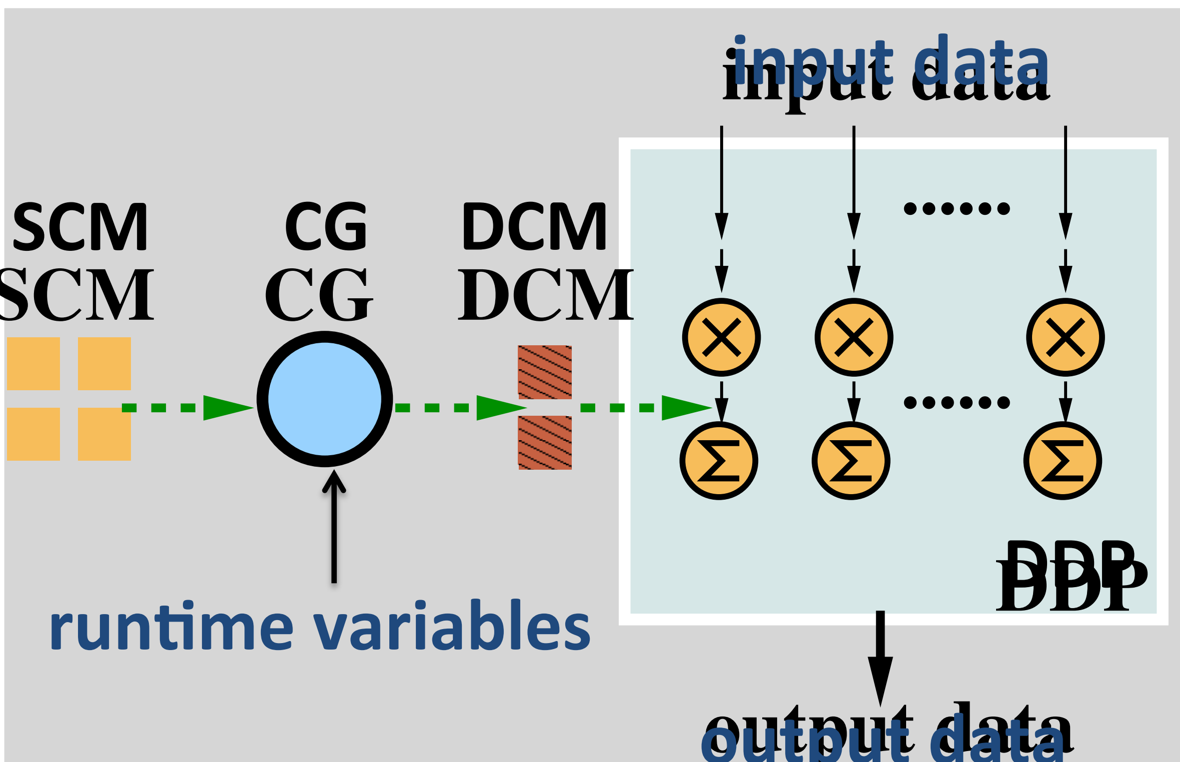


SCM: Static Configuration Memory

DDP: Dynamic Data-path

EURECA: on-chip configuration generation

- only active configuration is stored
- low reconfiguration time
- CGs are customisable, no fabrication overhead



DCM: Dynamic Configuration Memory

DDP: Dynamic Data-Path

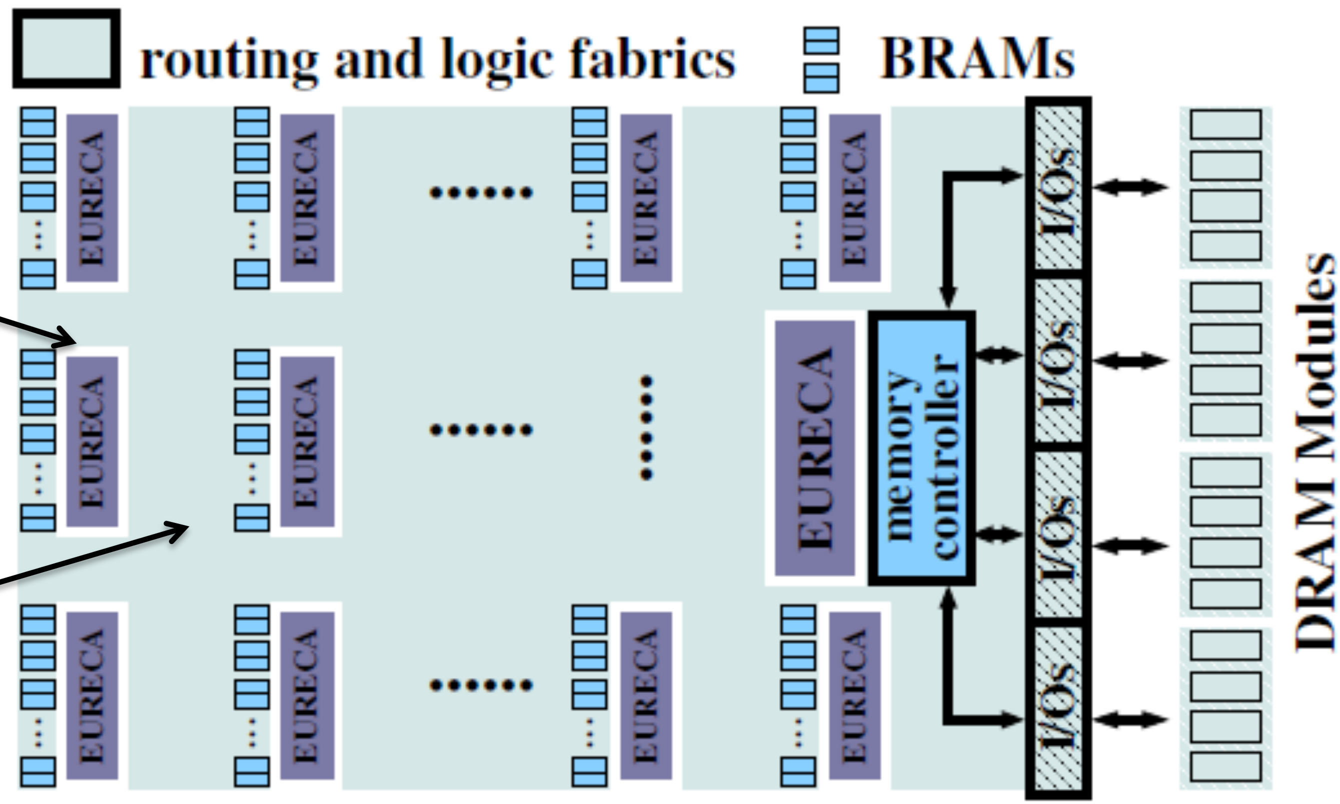
CG: Configuration Generator

EUREKA Architecture Overview

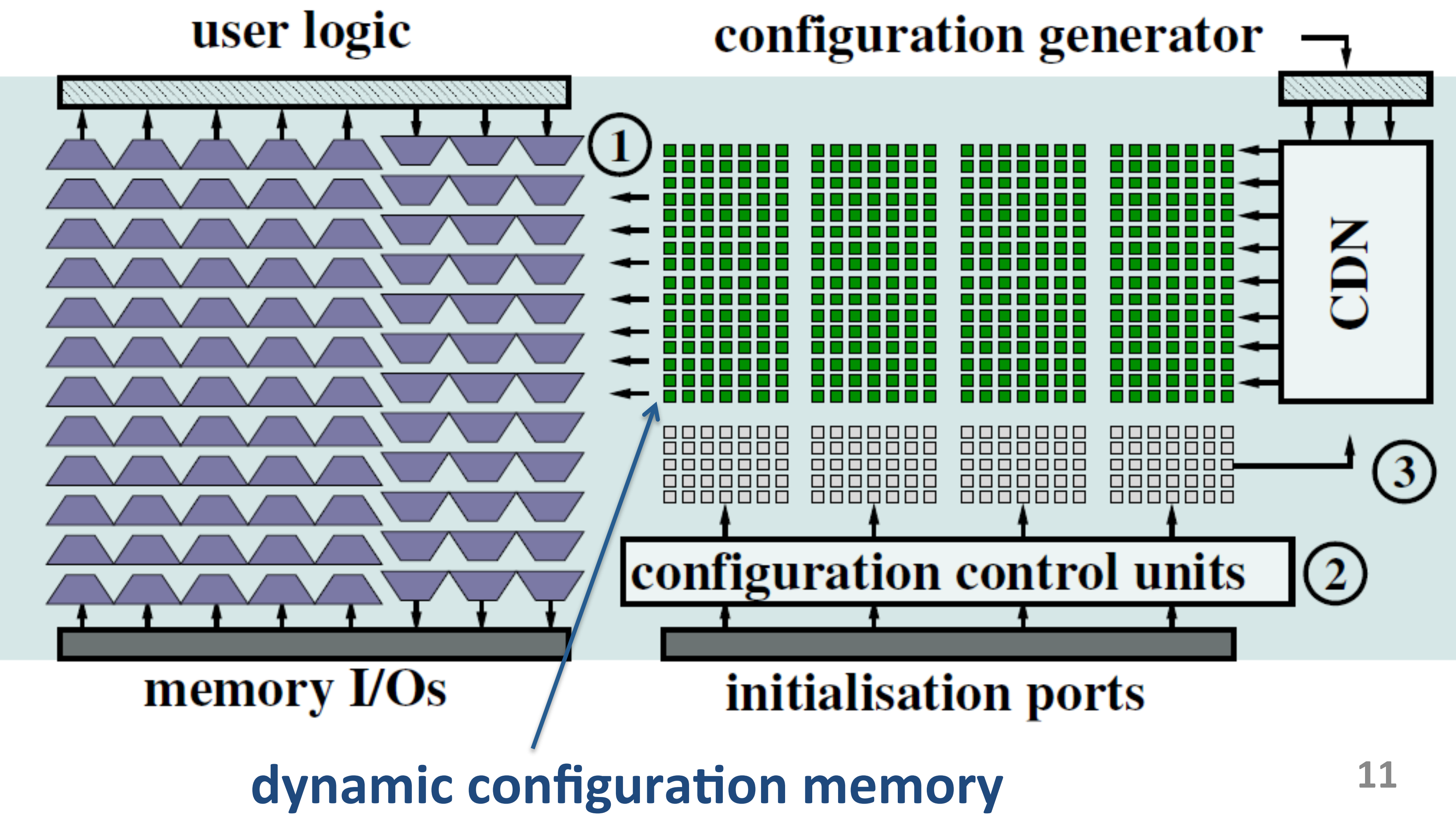
- address routing challenge: couple EURECA module to
  - a group of BRAM blocks
  - memory controller for off-chip DRAM

EURECA module: user logic to memory

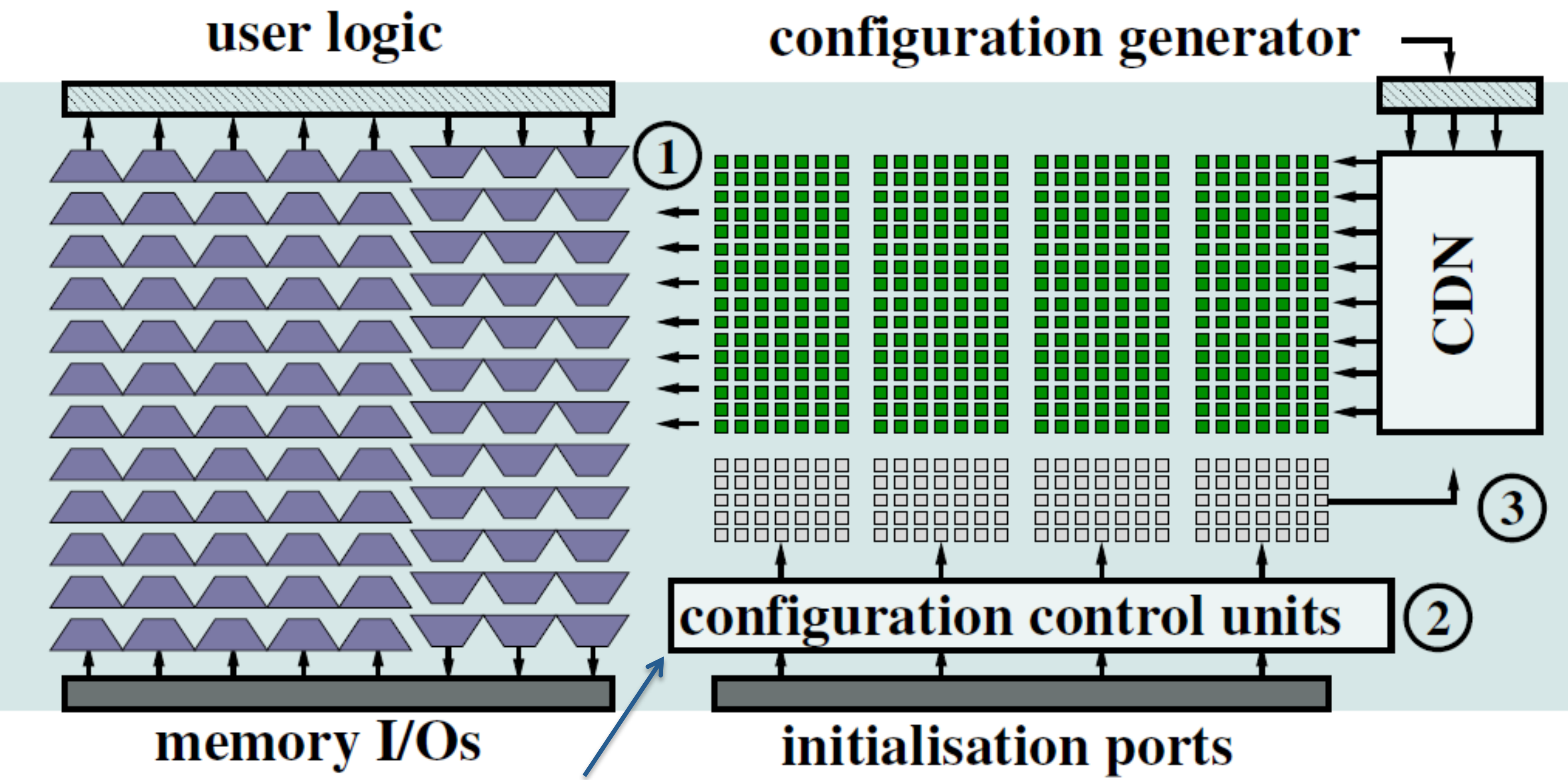
Configuration Generators in user logic



(1) Reconfigurable routing multiplexors

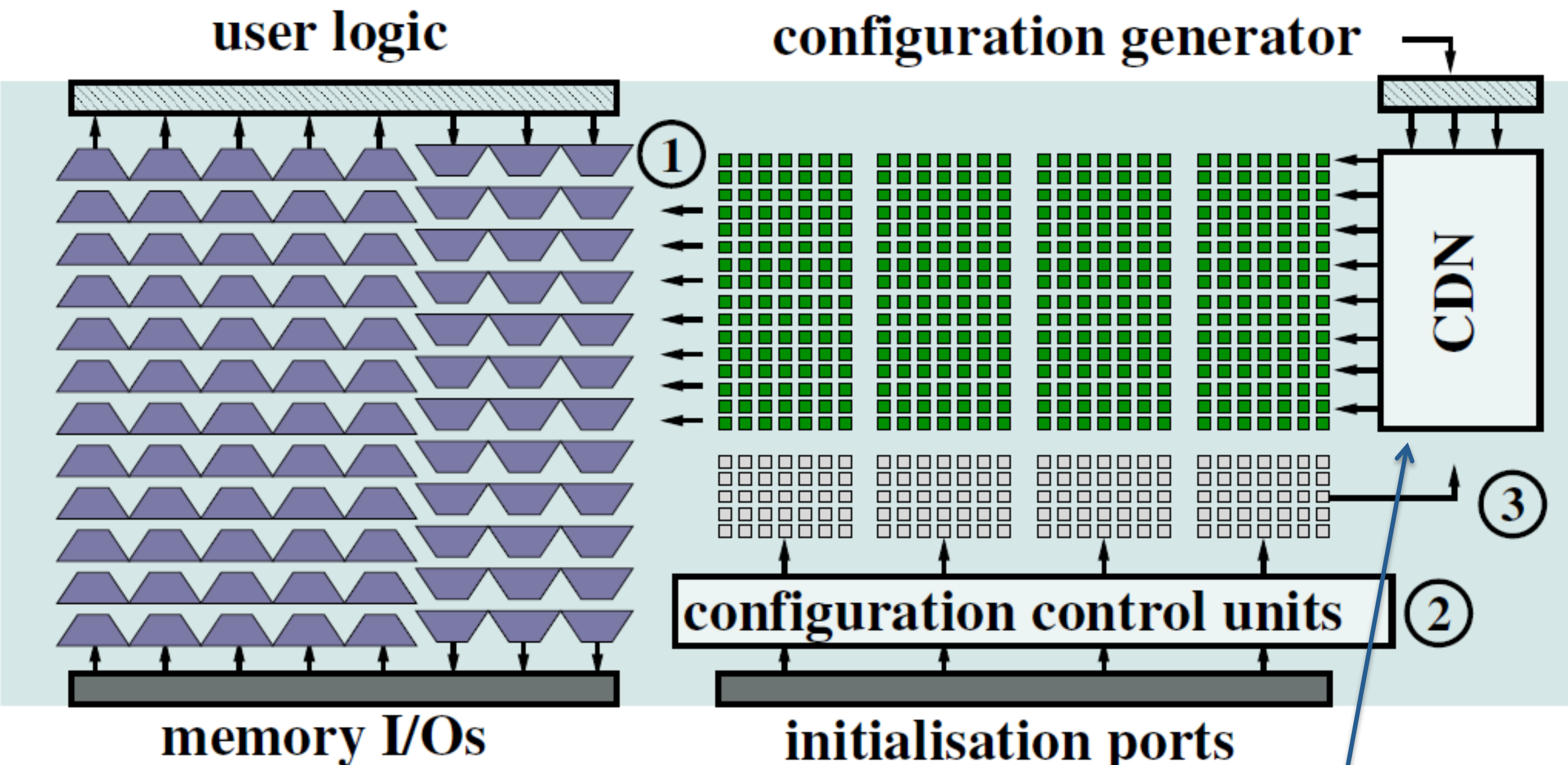


(2) Configuration control units



supports different modes: dynamic, static…

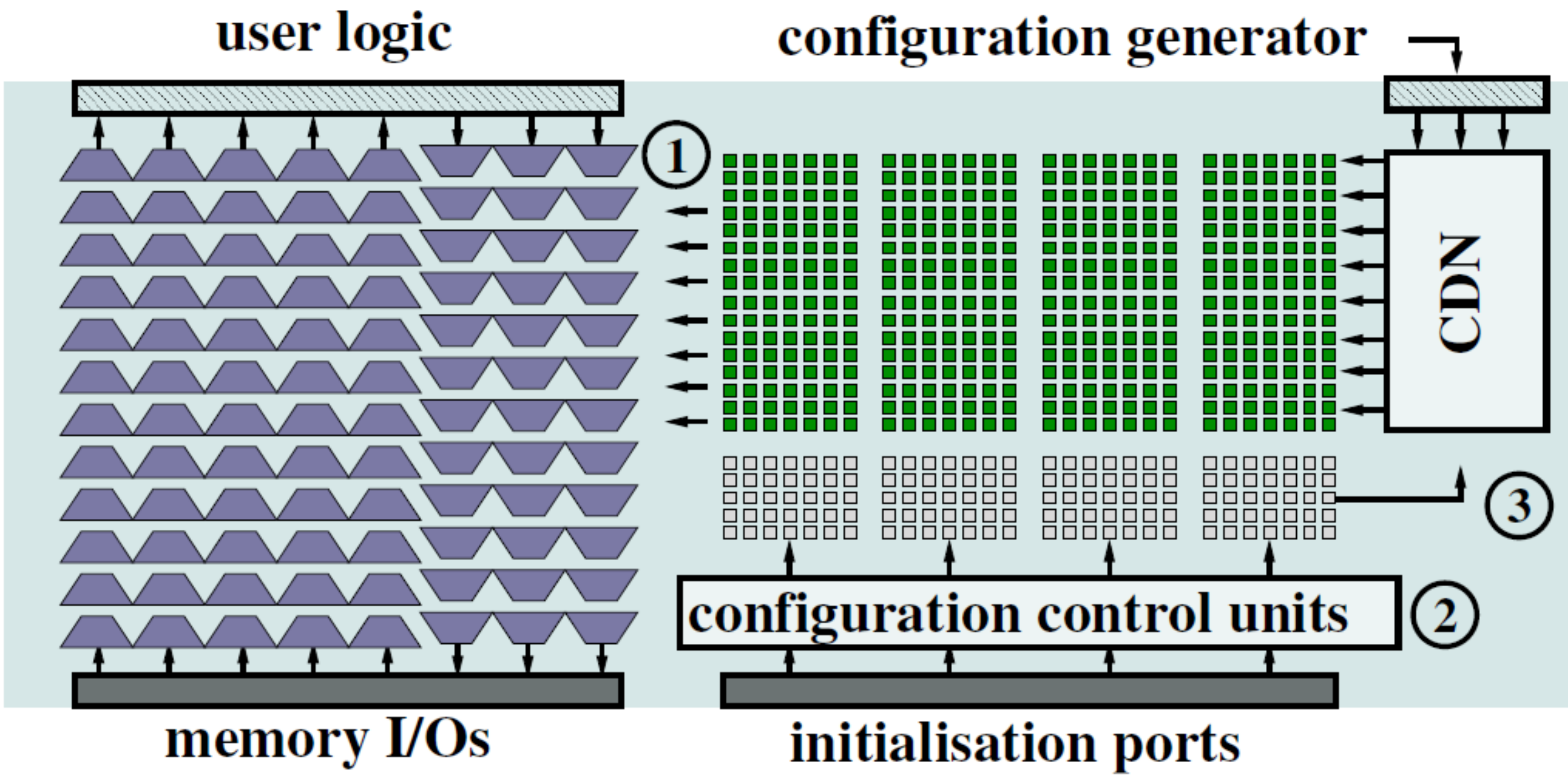
(3) CDN: Configuration distribution network



share generated configurations

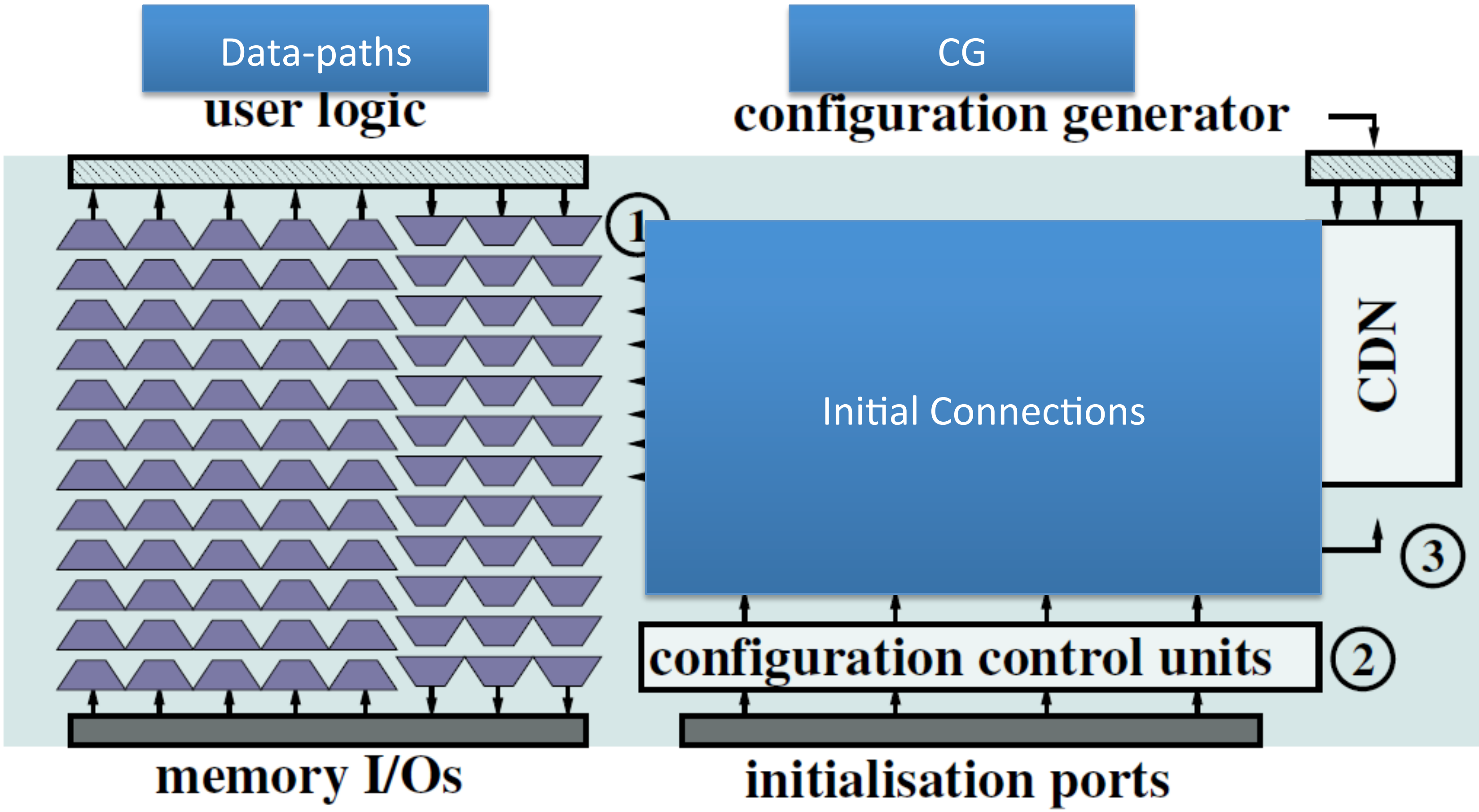
EURECA module: execution flow

Initial configuration

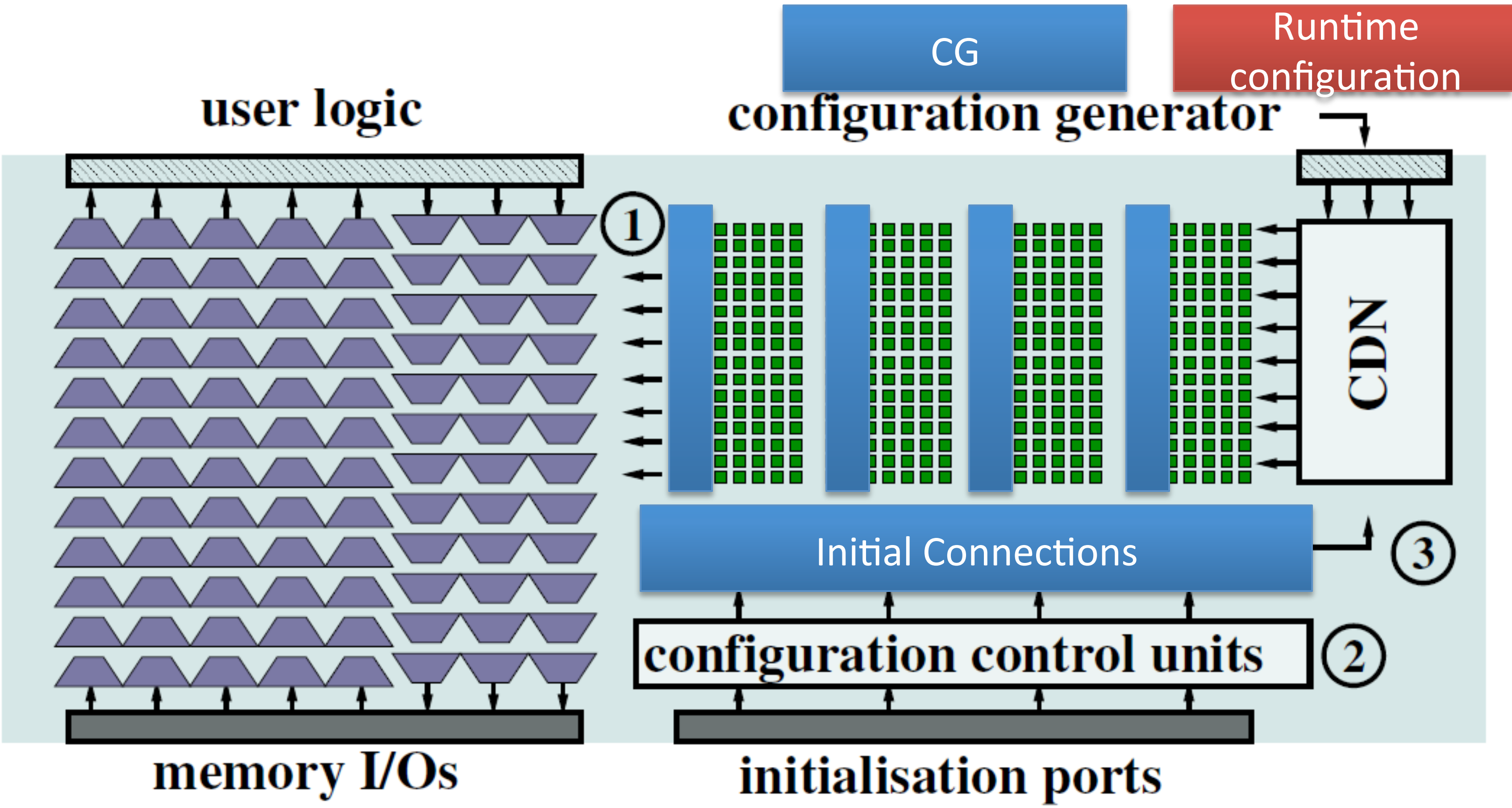


Initial configuration

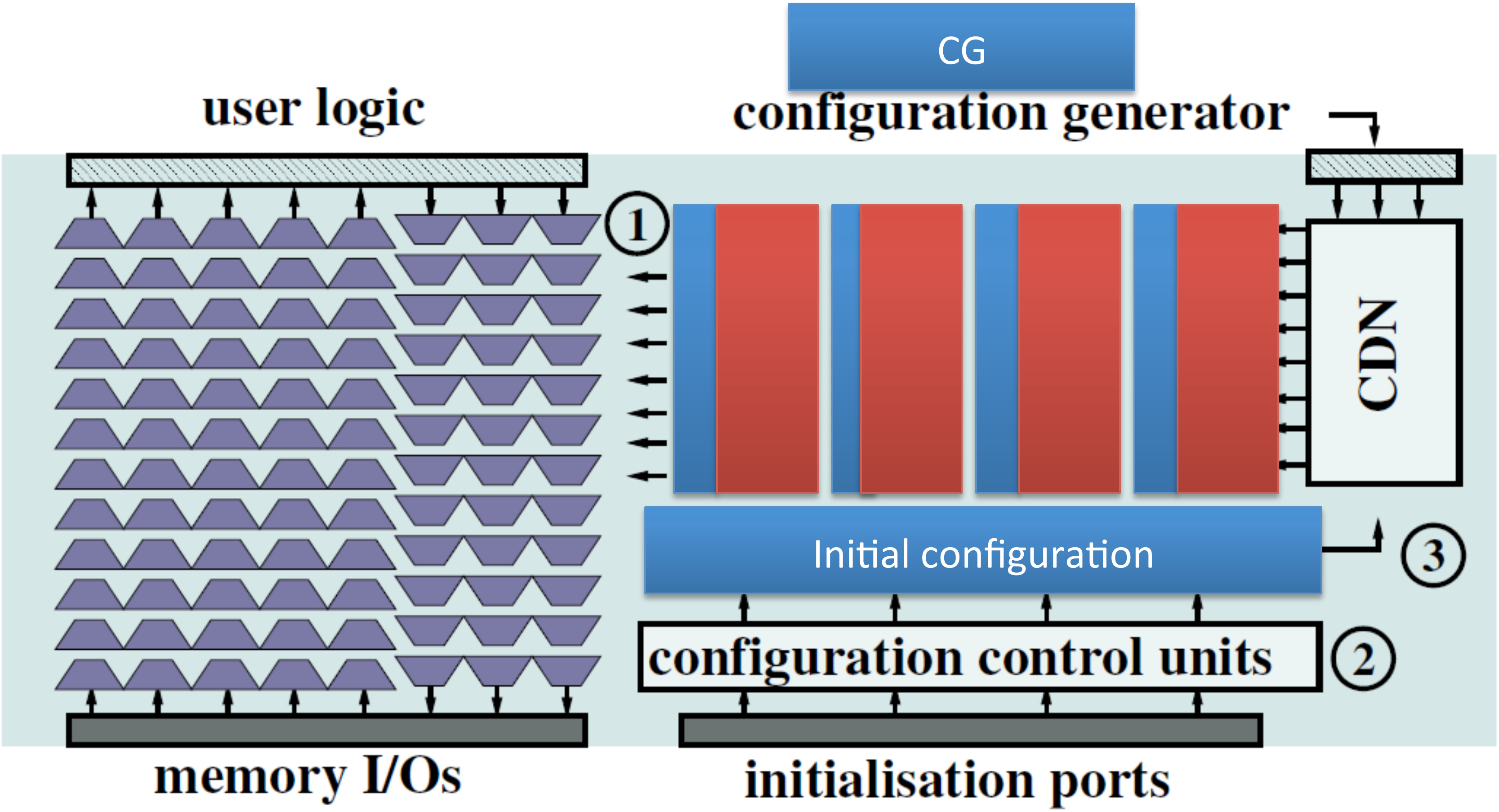
EURECA module: execution flow



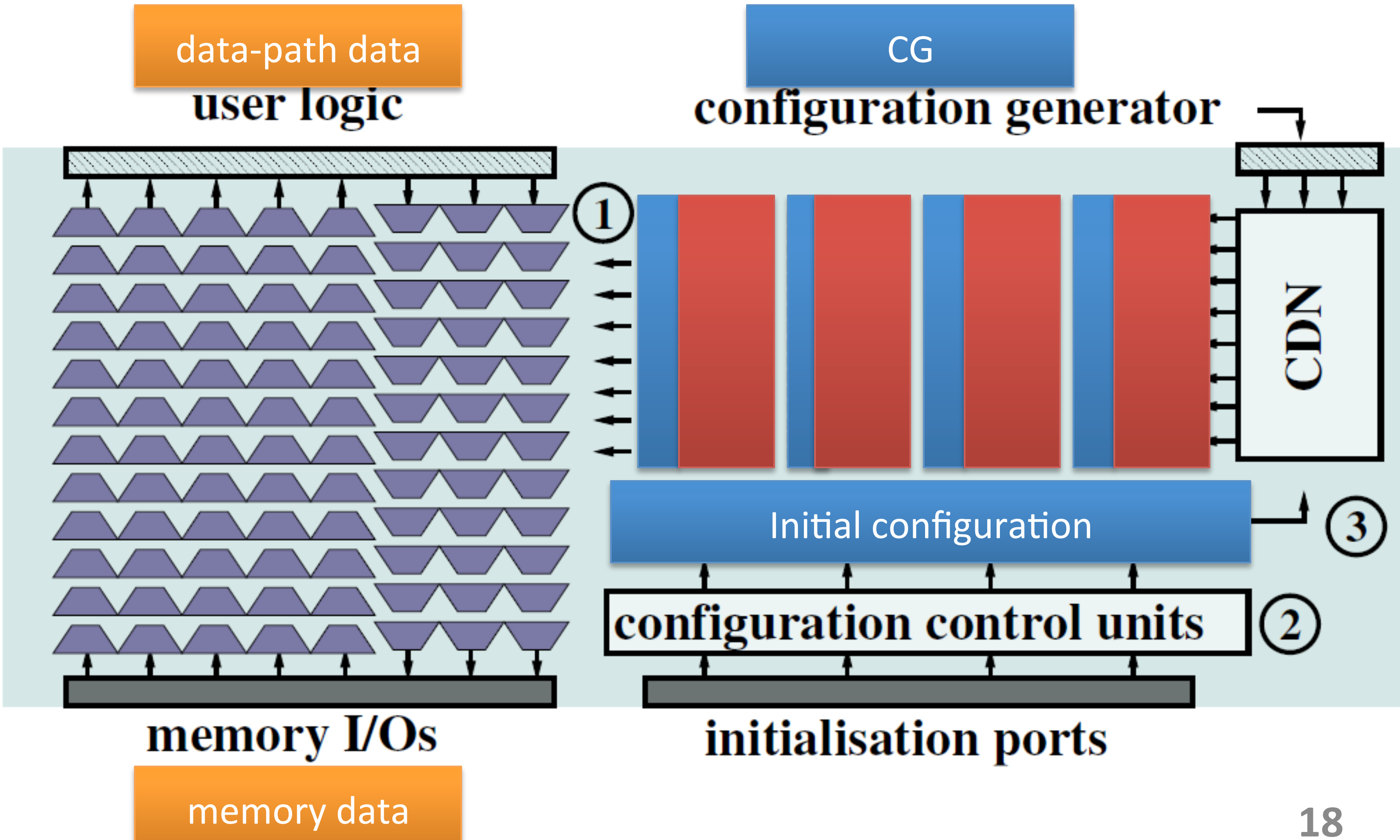
Run-time reconfiguration flow



Run-time reconfiguration flow

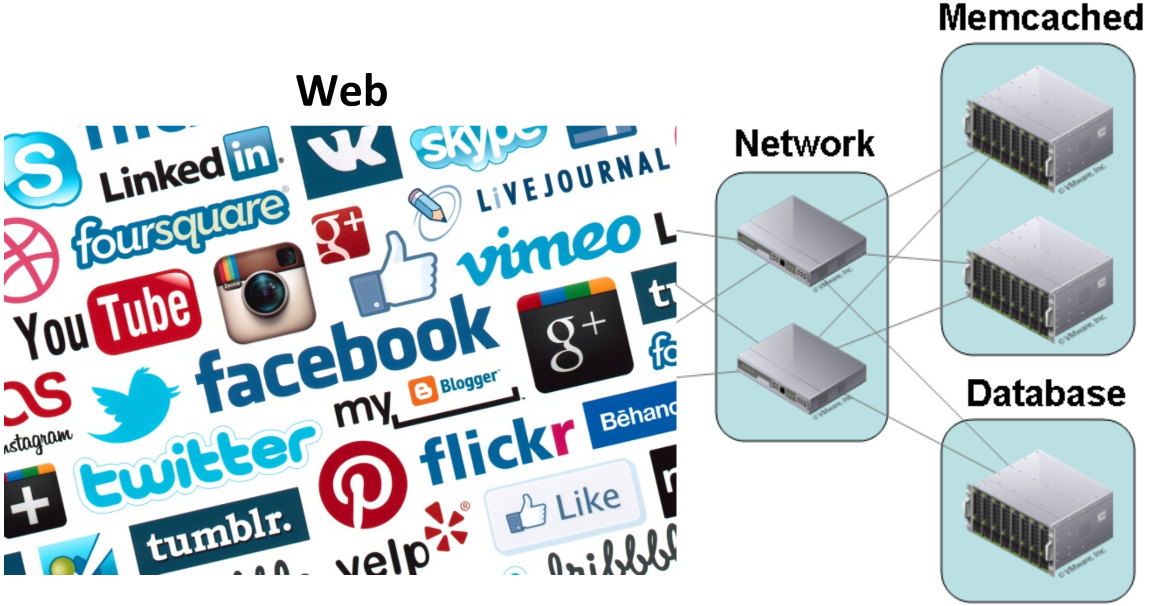


Run-time reconfiguration flow



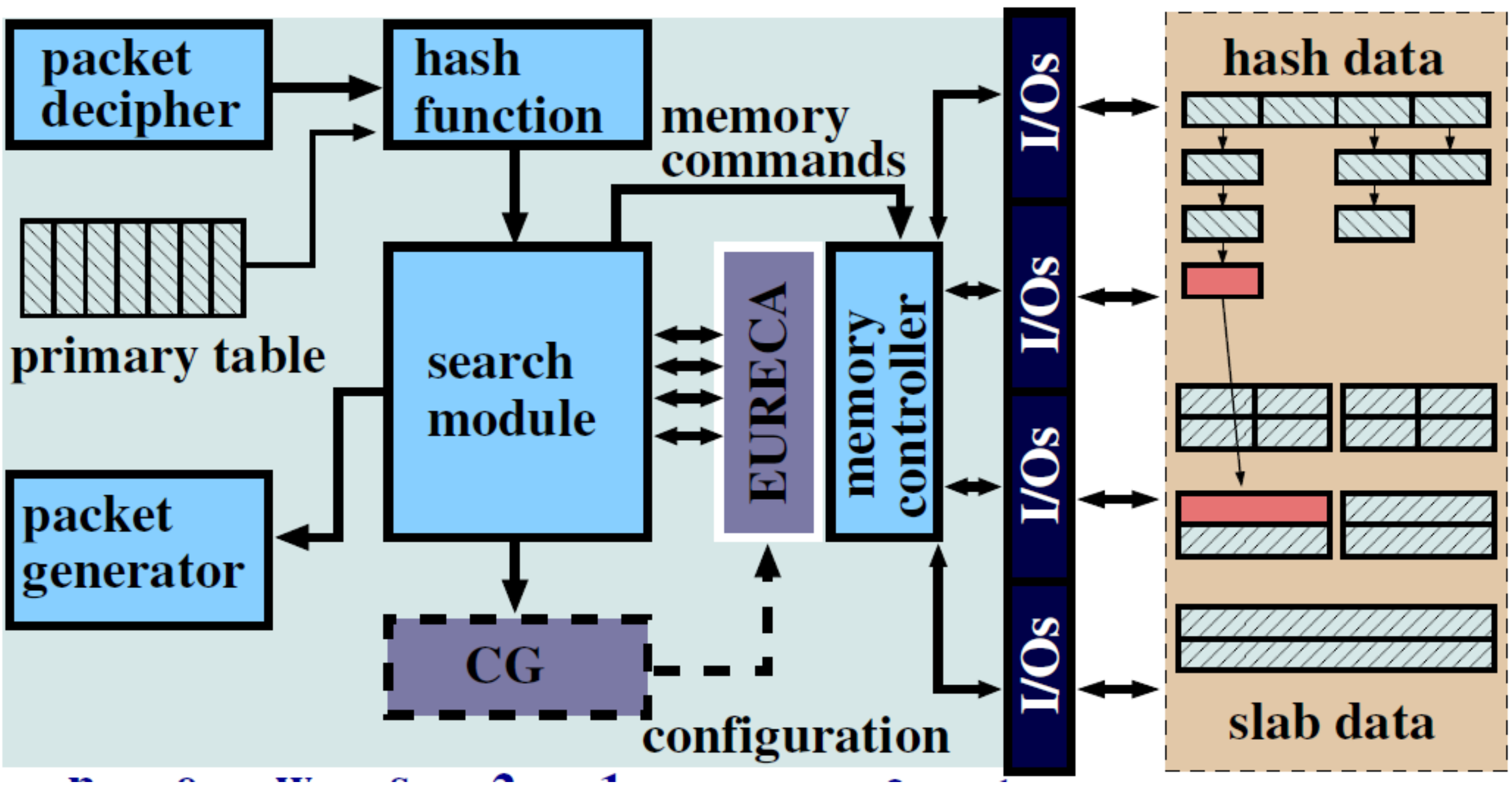
Case Study 1: Memcached

- widely used by Facebook, Twitter, Youtube…
- deployed between network and database
- using hash table to manage accessed items



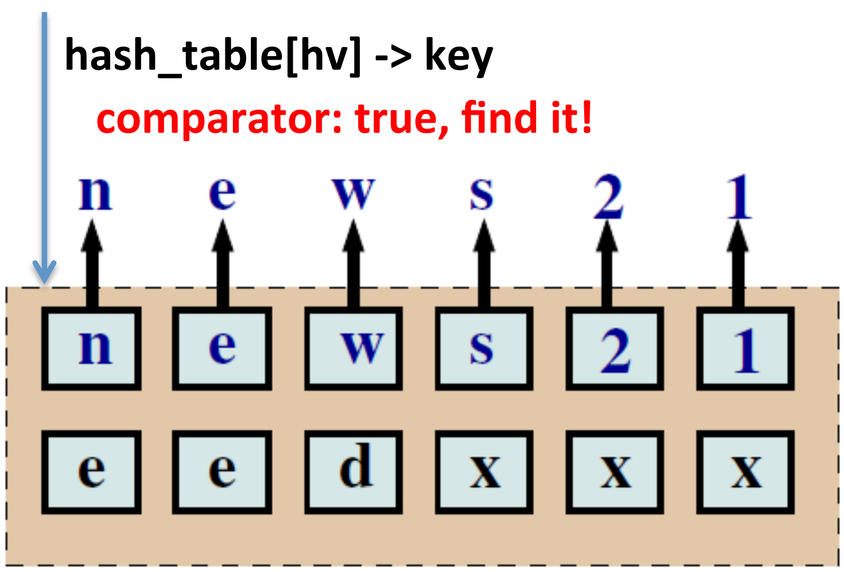
Memcached: architecture

- hash table + slab allocation
- request -> hashing -> primary table -> hash table



Challenge: unaligned data access

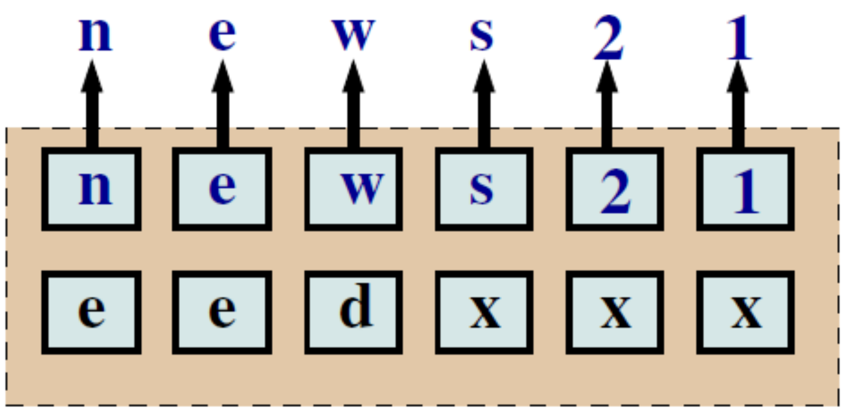
- variable key length in Memcached
- dynamic pointers pointing at unaligned data



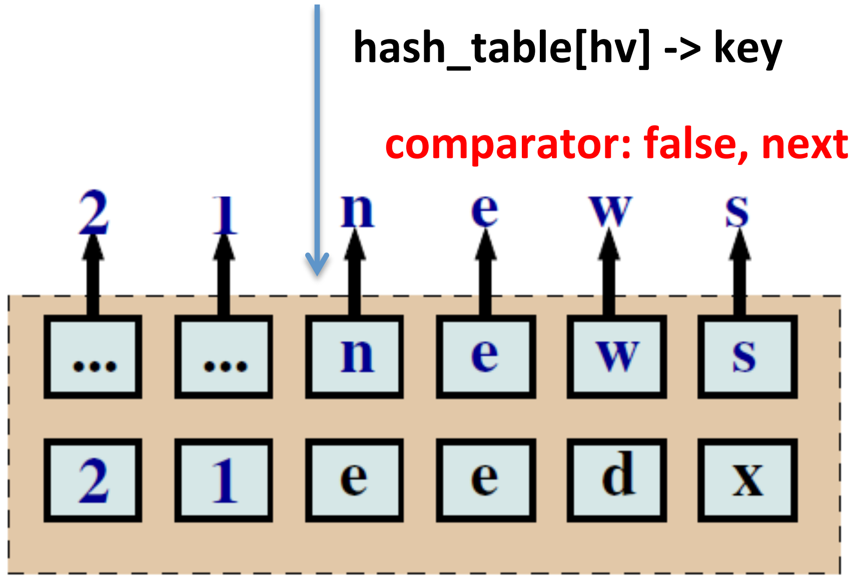
aligned data access

Challenge: unaligned data access

- variable key length in Memcached
- dynamic pointers pointing at unaligned data



aligned data access



unaligned data access:
wrong results

Memcached: comparing solutions

- static
- [4]: fixed key length
- [11]: ARM core + FPGA fabric
- EURECA

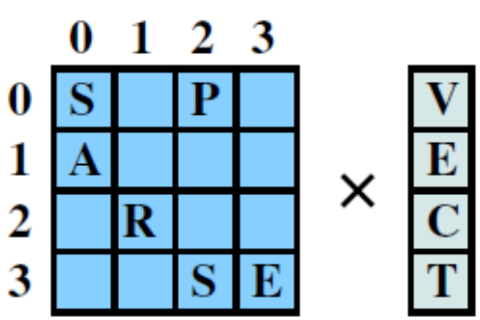
Table 1: Comparison of Memcached solutions.

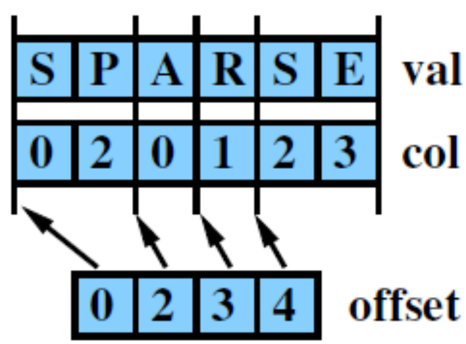
solution	complexity	functionality	throughput
static	N^2	full	N
[4]	N	fixed key length	$< N$
[11]	$N + C_{cpu}$	full	n/a
EURECA	$N + C_{eureca}$	full	N

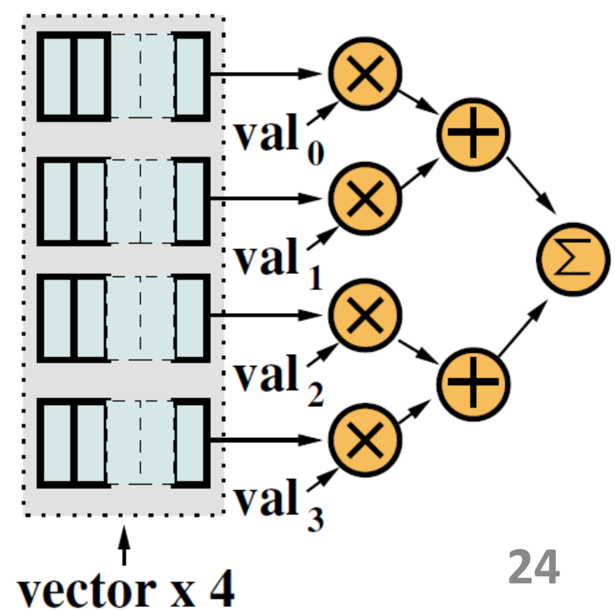
C_{eureca}: EURECA module area

Case Study 2: SpMV

- sparse Matrix-Vector Multiplication
- widely used in scientific computing
  - Linpack, finite-element, non-linear solver
- challenge: random access to vector data
- replicated vector memory
  - but large memory usage

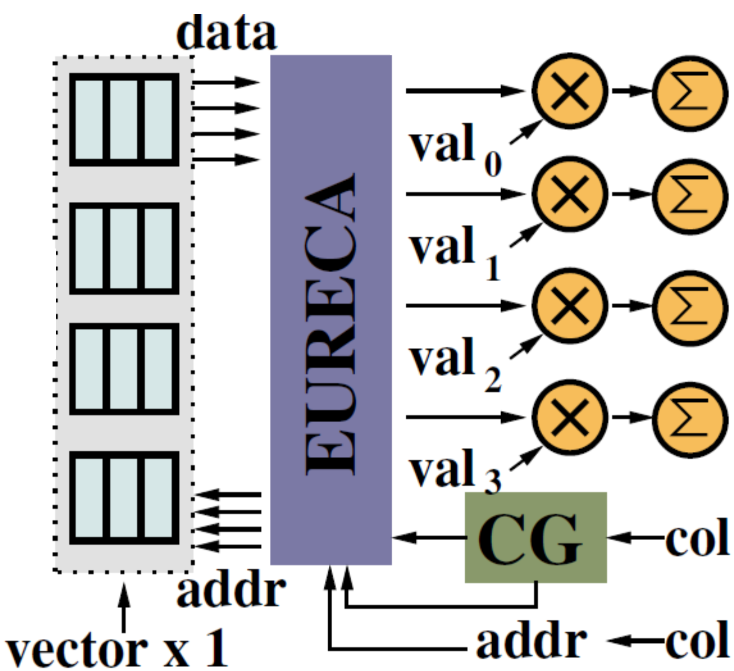






SpMV: comparing solutions

- [28]: replicated vector memory
- EURECA
  - shared vector memory
  - dynamic connections mapped to a EURECA module
  - configuration generated based on col input

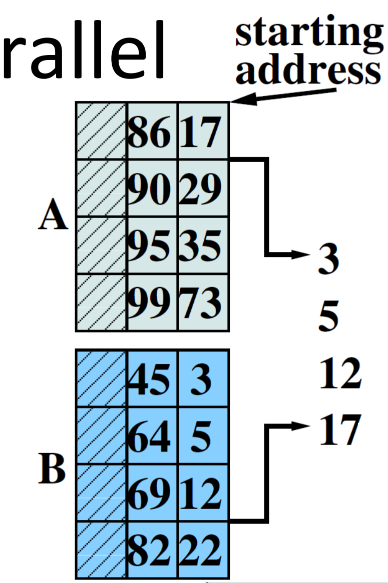


solution	complexity	vector size	efficiency
static	N^2	mem	85% ($\propto N$)
[28]	N	mem/N	42% ($\propto 1/N$)
EURECA	$N + C_{eureca}$	mem	85% ($\propto N$)

C_{eureca}: EURECA module area

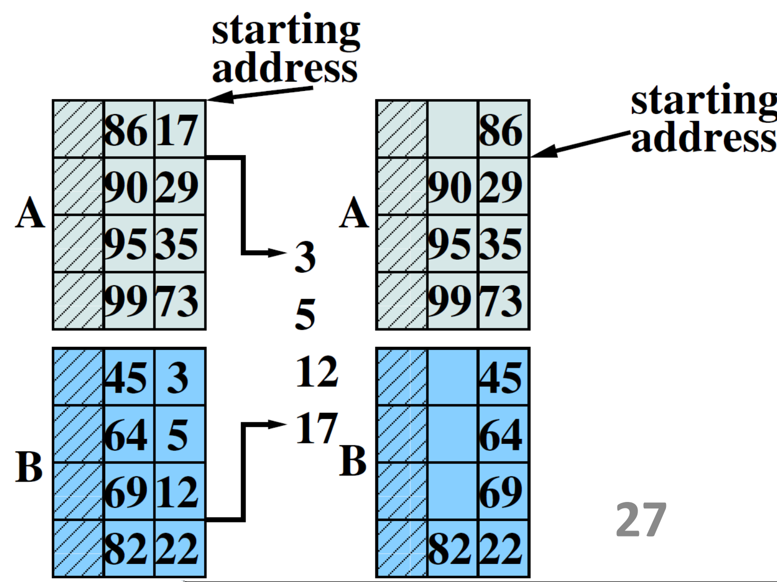
Case Study 3: Large-Scale Sorting

- extensively studied subject
  - in-memory database, database management
- challenge: result-dependent data access
  - sorting network only for small data
- when merging N=4 data in parallel
  - compare 2*4 data
  - commit the 4 smallest data



Case Study 3: Large-Scale Sorting

- extensively studied subject
  - in-memory database, database management
- challenge: result-dependent data access
  - sorting network only for small data
- when merging N=4 data in parallel
  - compare 2*4 data
  - commit the 4 smallest data
  - fetch the next 2*4 data
  - repeat previous steps



Large-scale sorting: comparing solutions

- sorting networks: small data
- parallel merger: N^2 complexity
- EURECA design: large data with complexity N

solution	complexity	data size	throughput
sorting network	$C \cdot N \log_2(N)$	small	N
merger	N^2	large	N
EURECA	$N + C_{eureca}$	large	N

C_{eureca}: EURECA module area

Experimental setup

- simulation
  - enhanced with EURECA infrastructure
  - EURECA module: based on Cadence Virtuoso in 65nm UMC technology
- synthesis environment
  - VTR 1.0
  - area and delay models from measuring cadence designs
  - Virtex-6 models from vendor specification

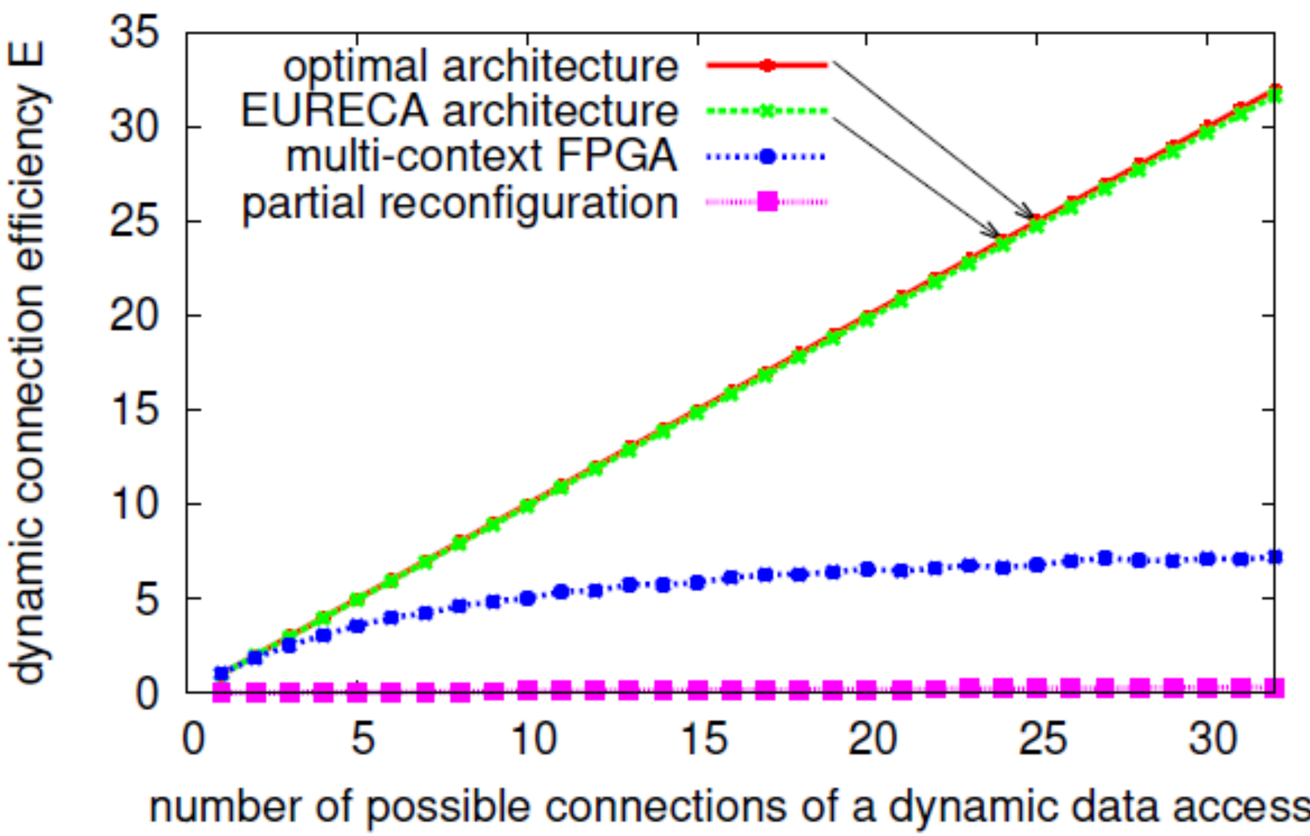
Dynamic connection efficiency

- optimal: unlimited R, no overhead
- EURECA: 1.17% area overhead
- multi-context FPGAs: limited by O_a
- partial reconfiguration: limited by O_t

$$E = \frac{R}{O} = \frac{R}{o_a \cdot o_t}$$

R: reconfigurability

O_a, area overhead,
O_t, reconfiguration time



EURECA: Performance

- compared with static designs in XC6V-SX475T FPGA
  - up to 1/15x design area
  - up to 2.2x clock speed
  - routable in EURECA architectures

	Memcached			SpMV			Large-scale Sorting		
	SDPs	baseline	EURECA	SDPs	baseline	EURECA	SDPs	baseline	EURECA
CLB	227	4399	234	459	3521	465	550	4875	561
DSP48x2	0	0	0	64	64	64	0	0	0
RAM36Kb	64	64	64	1024	1024	1024	64	64	64
EURECA	0	0	1	0	0	1	0	0	2
area[1] (10^6)	1.185	22.97	1.54	2.396	18.39	3.02	2.872	25.46	3.52
critical-path delay (ns)	6.7	13.94	6.46	6.54	12.74	6.17	9.51	11.56	9.51
area-delay product		**32.14x**	1x		12.57x	1x		8.792x	1x
routable[2]	✓	✗	✓	✓	✗	✓	✓	✗	✓
channel width	202	382[3]	211	215	317[3]	221	211	368[3]	204
throughput (per cycle)	128 bytes			32 partial results			32 sorted data		
enabled feature	flexible memory management			shared memory architecture			parallel merging		

Current and future work

- EURECA chip fabrication
- EURECA compiler
- Other applications
  - genomic sequence alignment
  - graph problems
- EURECA programming models
  - operation mapping
  - data access mapping
  - many-region communication

Summary: EURECA

- routing challenge: dynamic data access applications
  - multiplexors close to memory, configuration distribution
  - on-chip configuration generation
  - cycle-by-cycle reconfiguration
- experimental results: VTR + Cadence
  - small area overhead: 1% of XC6V-SX475T FPGA
- Memcached, sparse matric vector, large-scale sorting
  - up to 1/15x design area
  - up to 2.2x clock speed